



# Lecture: Image Resizing

Juan Carlos Niebles and Ranjay Krishna  
Stanford Vision and Learning Lab

# Today's agenda

- Image resizing
  - Seam carving
  - Dynamic programming
- Applications

# Display Devices



# Content Retargeting

The screenshot shows the BBC News homepage on a desktop computer. At the top, there's a navigation bar with links for News, Sport, Weather, Travel, TV, Radio, More, and a search bar. Below the navigation is a large image of a person speaking at a podium. A red banner across the image reads "TOP NEWS STORY" and "Tunisia leaders quit ruling party". Below the banner, a brief summary of the story is provided. To the right of the main story, there are several other news cards: "News" (Duvalier taken to court in Haiti), "Sport" (Live - Tuesday football about 2 hours ago), "Business" (Market Data for Tuesday, 18 Jan 2011), "World Service" (News in 32 languages with links to Arabic, Chinese, Russian, etc.), and "TV Channels" (List of BBC channels). On the left side, there are additional sections for "Sci & Environment" (India aims for tidal power first) and "Entertainment" (King's Speech leads Bafta field).

PC

The screenshot shows the BBC News homepage on a mobile device (iPhone). The layout is similar to the desktop version but adapted for a smaller screen. It includes the top navigation bar, the "TOP NEWS STORY" banner with the Tunisia story, and various news cards. The "Spotlight" card features a video thumbnail of Michael S Malone. The "Business" card shows market data for Dow Jones, Nasdaq, FTSE 100, Dax, and Cac 40. The "World Service" card lists news in multiple languages. The "TV Channels" card shows a list of BBC channels. The overall design is responsive, with some sections like "Business" and "World Service" being scaled down or removed to fit the phone's screen.

iPhone



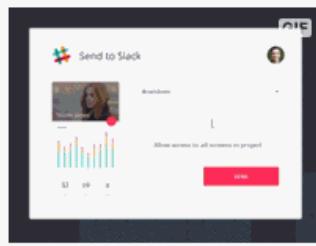
# Page Layout

working on? Dribbble is show and tell for designers.

Learn more

Sign up

Popular ▾   Shots ▾   Now ▾



InVision



Vladimir Marchukov



Brian Steely



CHOOSING COLOURS  
FOR YOUR NEXT ICON SET

796 views, 5 comments, 135 likes

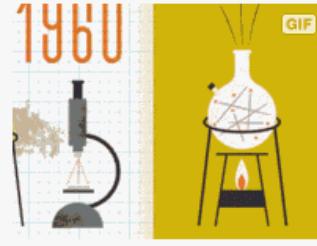
Justas Galaburda



Lionel Durimel



Szende Brassai / Adline



Dustin Wallace



1,553 views, 8 comments, 126 likes

Vladimir Babic



# Simple Media Retargeting Operators

Letterboxing



# Content-aware Retargeting Operators

Content-aware



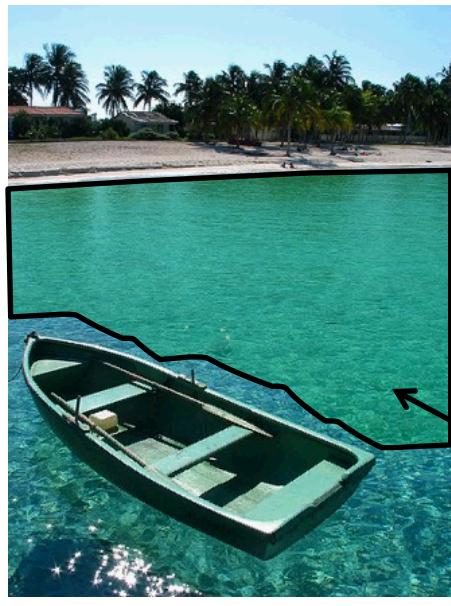
"Important" content



Content-oblivious



# Content-aware Retargeting



Input



Scale



Crop



Content-aware

“less-Important”  
content

# Image Retargeting

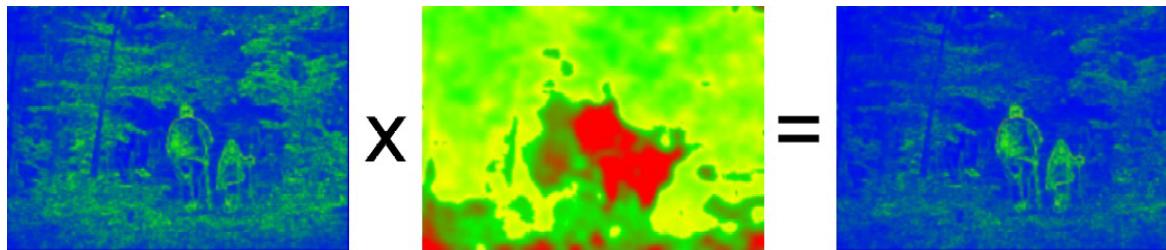
- Problem statement:
  - Input Image  $I$   $n \times m$ , and new size  $n' \times m'$
  - Output Image  $I'$  of size  $n' \times m'$  which will be “good representative” of the original image  $I$
- To date, no agreed definition, or measure, as to what a good representative is in this context!

# Image/Video Retargeting

- In large, we would expect retargeting to:
  1. Adhere to the geometric constraints (display/aspect ratio)
  2. Preserve the important *content* and *structures*
  3. Limit *artifacts*
- Very Ill-posed!
  - How do we define important? Is there a universal ground truth?
  - Would different viewers think the same about a retargeted image?
  - What about artistic impression in the original content?

# Importance (Saliency) Measures

- A function  $S: p \rightarrow [0,1]$



Wang et al. 2008

- More sophisticated: attention models, eye tracking (gazing studies), face detectors, ...

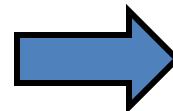
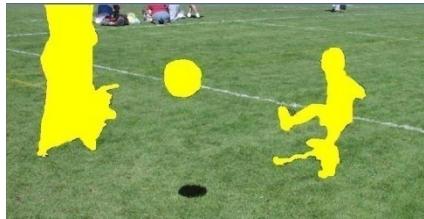


Judd et al. ICCV09 *Learning to predict where people look*

# General Retargeting Framework

1. Define an energy function  $E(I)$   
(interest, importance, saliency)

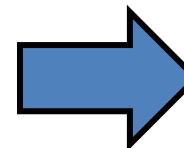
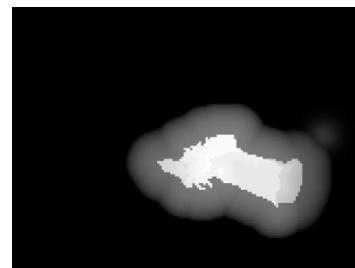
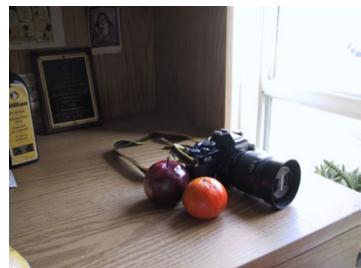
2. Use some operator(s) to  
change the image  $I$



*Recompose*



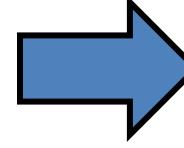
Setlur et al.  
[2005]



*Crop*



Santella et  
al. [2005]



*Warp*



Gal et al.  
[2006]

# Previous Retargeting Approaches

- Optimal Cropping Window



- For videos: “Pan and scan”  
Still done manually in the movie industry



Liu and Gleicher, *Video Retargeting: Automating Pan and Scan* (2006)

# Cropping



# Seam Carving

- Assume  $m \times n \rightarrow m \times n'$ ,  $n' < n$  (summarization)

# Seam Carving

- Assume  $m \times n \rightarrow m \times n'$ ,  $n' < n$  (summarization)
- Basic Idea: remove unimportant pixels from the image
  - Unimportant = pixels with less “energy”

$$E_1(\mathbf{I}) = \left| \frac{\partial}{\partial x} \mathbf{I} \right| + \left| \frac{\partial}{\partial y} \mathbf{I} \right|.$$

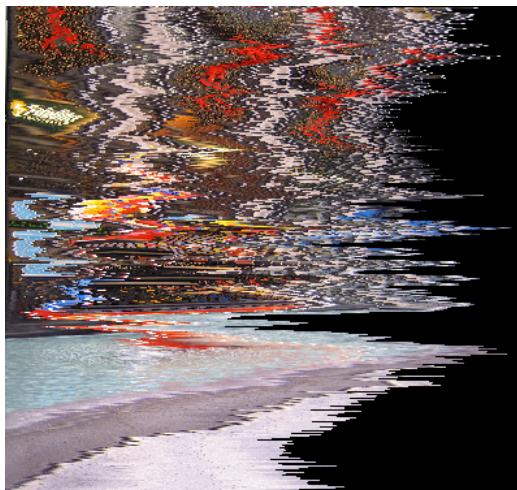
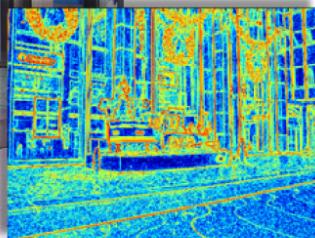
# Seam Carving

- Assume  $m \times n \rightarrow m \times n'$ ,  $n' < n$  (summarization)
- Basic Idea: remove unimportant pixels from the image
  - Unimportant = pixels with less “energy”

$$E_1(\mathbf{I}) = \left| \frac{\partial}{\partial x} \mathbf{I} \right| + \left| \frac{\partial}{\partial y} \mathbf{I} \right|.$$

- Intuition for gradient-based energy:
  - Preserve strong contours
  - Human vision more sensitive to edges – so try remove content from smoother areas
  - Simple enough for producing some nice results

# Pixel Removal



Optimal



Least-energy pixels  
(per row)

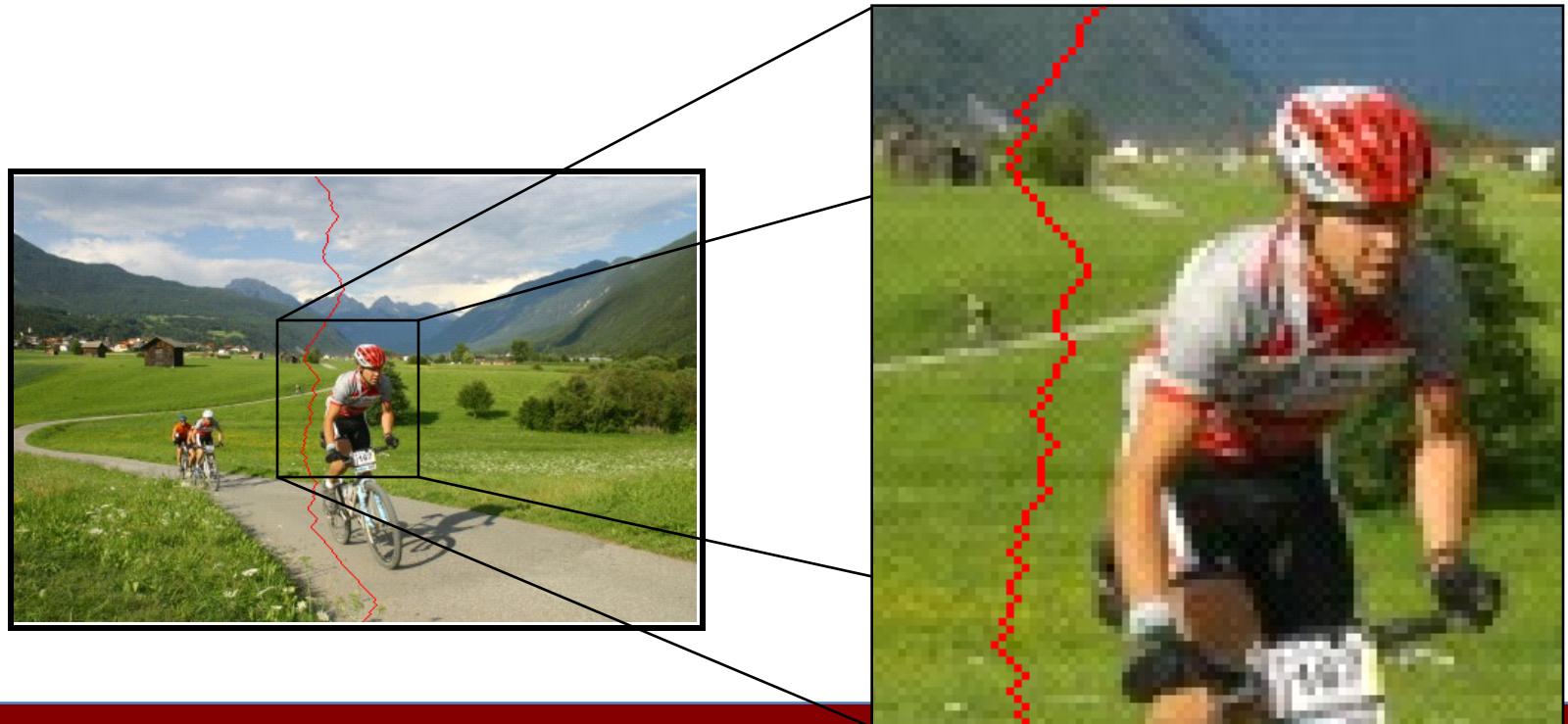


Least-energy columns

# A Seam

- A connected path of pixels from top to bottom (or left to right). Exactly one in each row

$$\mathbf{s}^x = \{s_i^x\}_{i=1}^n = \{(x(i), i)\}_{i=1}^n, \text{ s.t. } \forall i, |x(i) - x(i - 1)| \leq 1$$

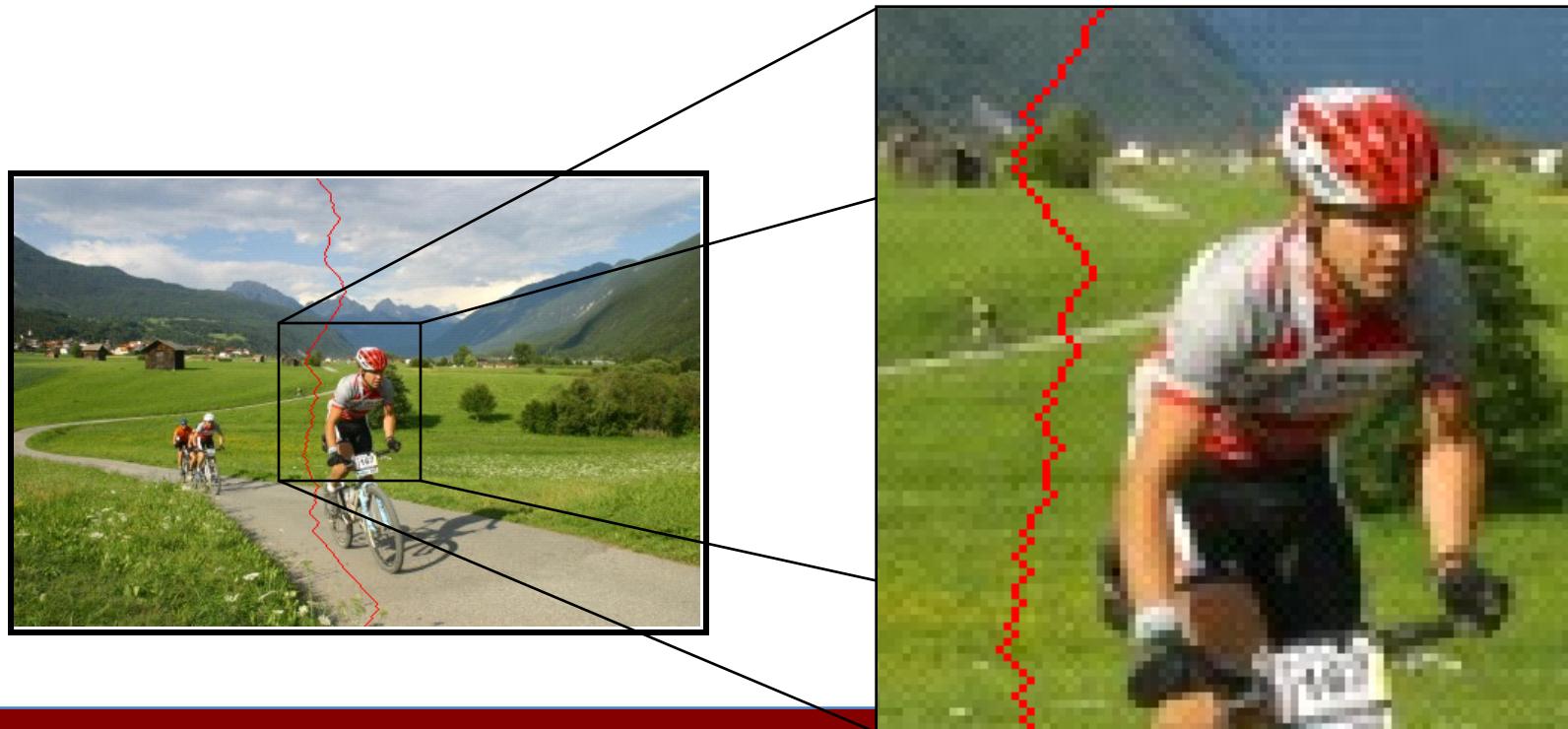


# A Seam

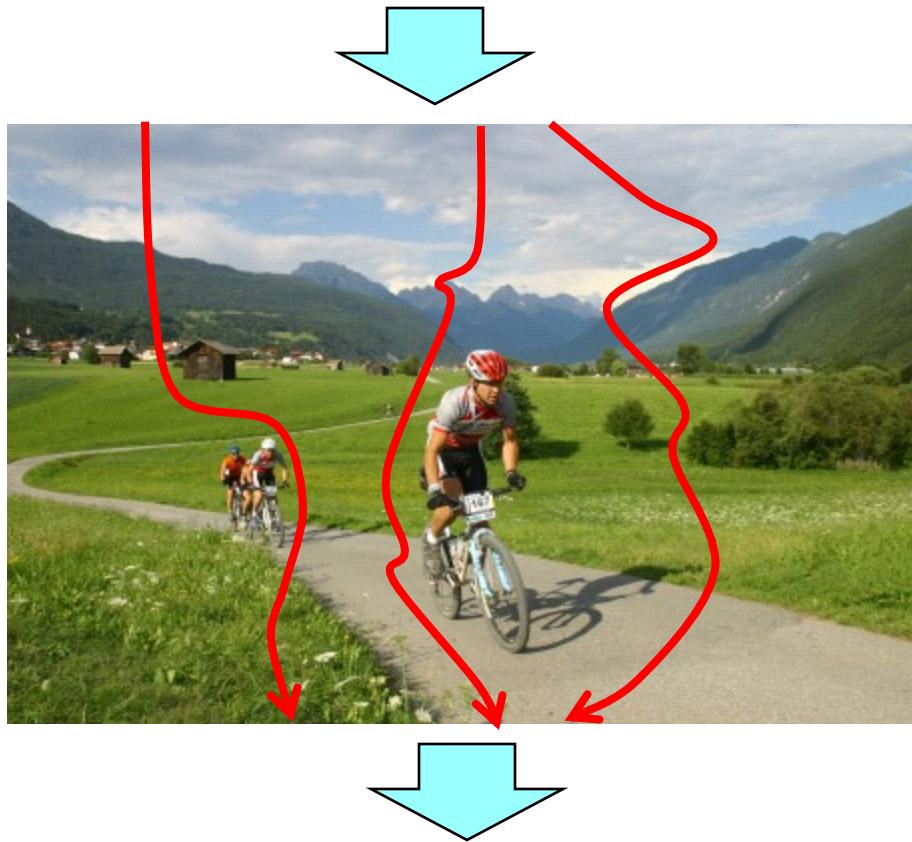
- A connected path of pixels from top to bottom (or left to right). Exactly one in each row

$$\mathbf{s}^x = \{s_i^x\}_{i=1}^n = \{(x(i), i)\}_{i=1}^n, \text{ s.t. } \forall i, |x(i) - x(i-1)| \leq 1$$

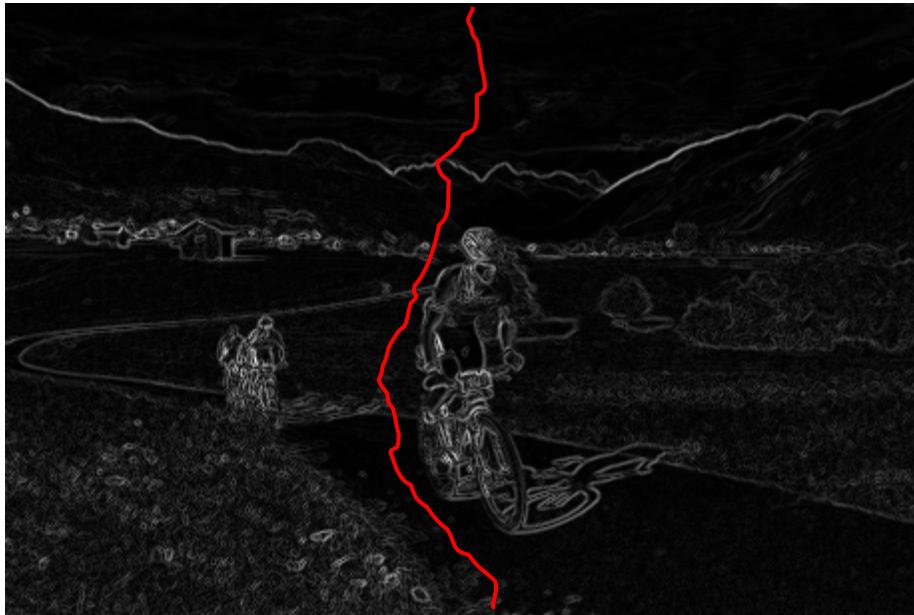
$$\mathbf{s}^y = \{s_j^y\}_{j=1}^m = \{(j, y(j))\}_{j=1}^m, \text{ s.t. } \forall j, |y(j) - y(j-1)| \leq 1$$



# Finding the Seam?



# The Optimal Seam



$$E(\mathbf{I}) = \left| \frac{\partial}{\partial x} \mathbf{I} \right| + \left| \frac{\partial}{\partial y} \mathbf{I} \right| \rightarrow s^* = \arg \min_s E(s)$$

# The Optimal Seam

- The recursion relation

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

- Can be solved efficiently using dynamic programming in  $O(s \cdot n \cdot m)$   
( $s=3$  in the original algorithm)

# Dynamic Programming

- Invariant property:
  - $M(i,j)$  = minimal cost of a seam going through  $(i,j)$  (satisfying the seam properties)

5	8	12	3
9	2		

Diagram illustrating a seam in a 4x4 grid. The top row contains values 5, 8, 12, 3. The second row contains values 9, 2, empty, empty. Red arrows point from the value 5 in the first column of the top row to the value 9 in the second row, and from the value 8 in the second column of the top row to the value 2 in the second row.

5	8	12	3
4	2	3	9
7	3	4	2
5	5	7	8

Energy -  $E(i,j)$

# Dynamic Programming

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

5	8	12	3
9	2+5		

5	8	12	3
4	2	3	9
7	3	4	2
5	5	7	8

Energy -  $E(i, j)$

# Dynamic Programming

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

5	8	12	3
9	7	3+3	

5	8	12	3
4	2	3	9
7	3	4	2
5	5	7	8

Energy -  $E(i, j)$

# Dynamic Programming

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	8+8

5	8	12	3
4	2	3	9
7	3	4	2
5	5	7	8

Energy -  $E(i, j)$

# Searching for Minimum

- Backtrack (can store choices along the path, but do not have to)

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	16



# Backtracking the Seam

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	16

# Backtracking the Seam

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	16

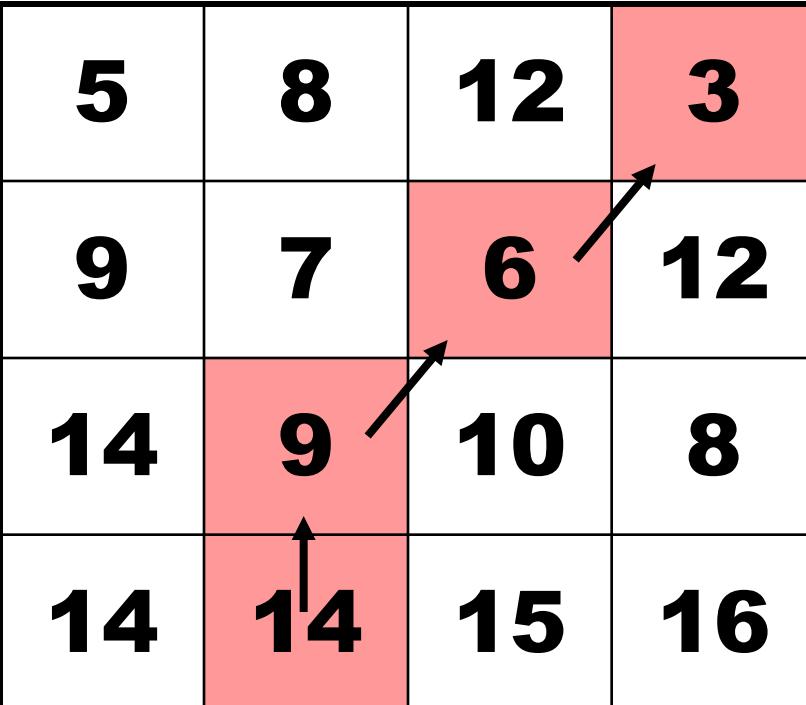
The diagram shows a 4x4 grid of numbers. The values are as follows:

- Row 1: 5, 8, 12, 3
- Row 2: 9, 7, 6, 12
- Row 3: 14, 9, 10, 8
- Row 4: 14, 14, 15, 16

A double-headed vertical arrow is positioned below the first column of the third row (value 14), indicating it is being considered or has been previously selected. An arrow points from the value 9 in the second row to the third column of the same row, likely indicating a move or comparison.

# Backtracking the Seam

5	8	12	3
9	7	6	12
14	9	10	8
14	14	15	16

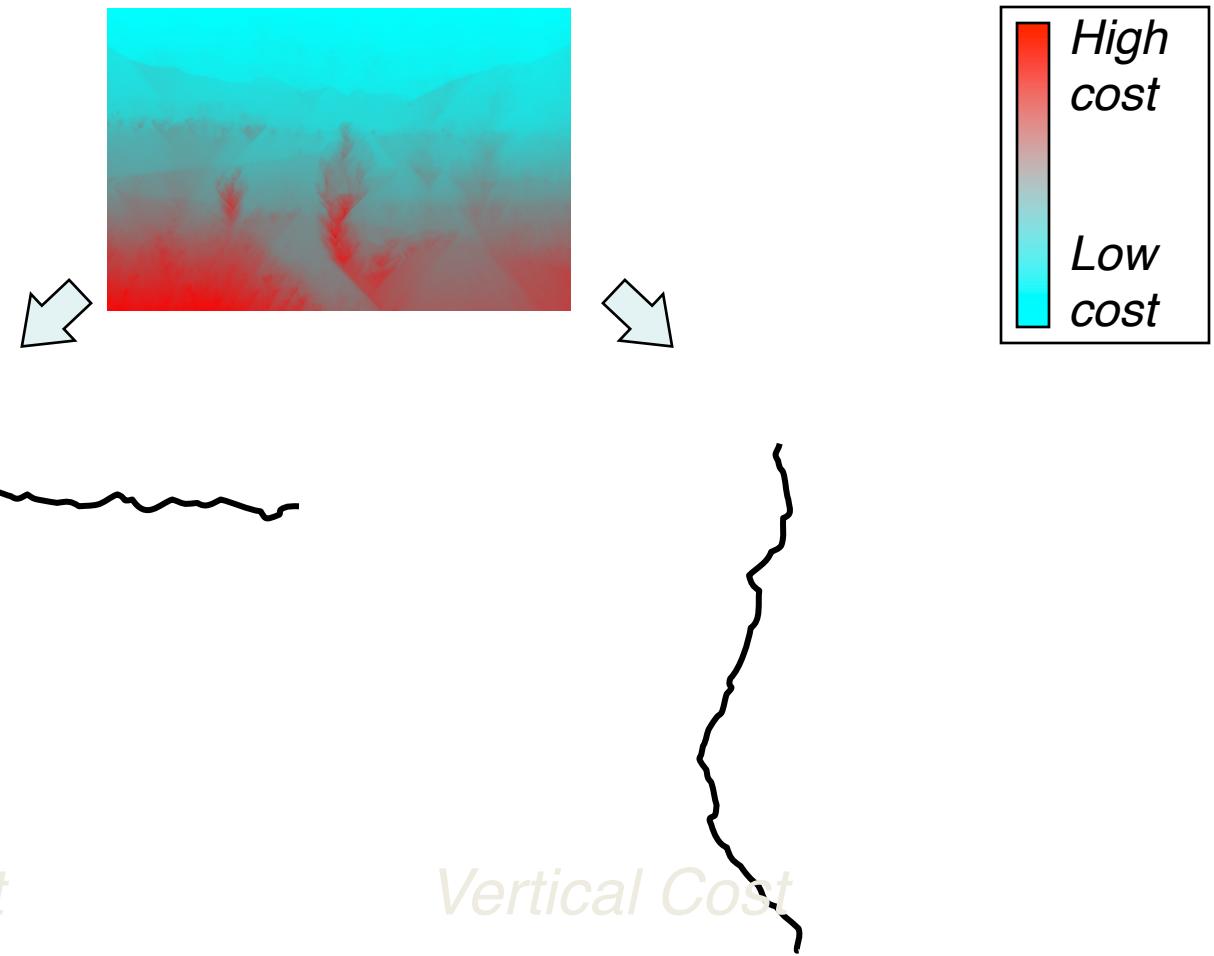


The diagram illustrates a 4x4 grid of numbers. The values are as follows:

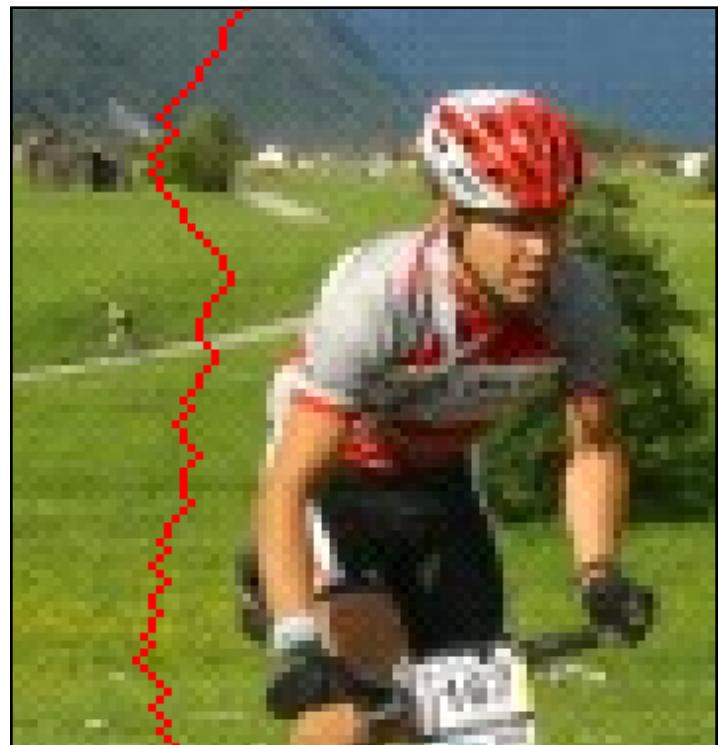
- Row 1: 5, 8, 12, 3
- Row 2: 9, 7, 6, 12
- Row 3: 14, 9, 10, 8
- Row 4: 14, 14, 15, 16

Cells (1,4), (2,3), (3,2), and (4,1) are highlighted with red boxes. Arrows point from the bottom-left cell (14) up to (3,2) and from (3,2) right to (2,3).

# H & V Cost Maps



# Seam Carving



# The Seam-Carving Algorithm

```
SEAM-CARVING(im,n') // size(im) = mxn
```

1. Do  $(n-n')$  times
  - 2.1.  $E \leftarrow$  Compute energy map on im
  - 2.2.  $s \leftarrow$  Find optimal seam in E
  - 2.3.  $im \leftarrow$  Remove s from im
2. Return im

- For vertical resize: transpose the image
- Running time:
  - 2.1  $O(mn)$
  - 2.2  $O(mn)$
  - 2.3  $O(mn)$

$\rightarrow O(dmn)$   $d=n-n'$

# Changing Aspect Ratio



# Changing Aspect Ratio



*Original*



*Seam Carving*



*Scaling*

# Changing Aspect ratio



*Cropping*



*Seams*



*Scaling*

# Changing Aspect Ratio



**Original**



**Retarget**



*Scaling*

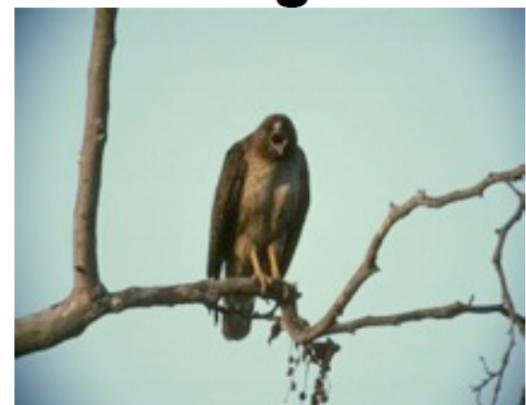
# Changing Aspect Ratio



**Original**



**Retarget**

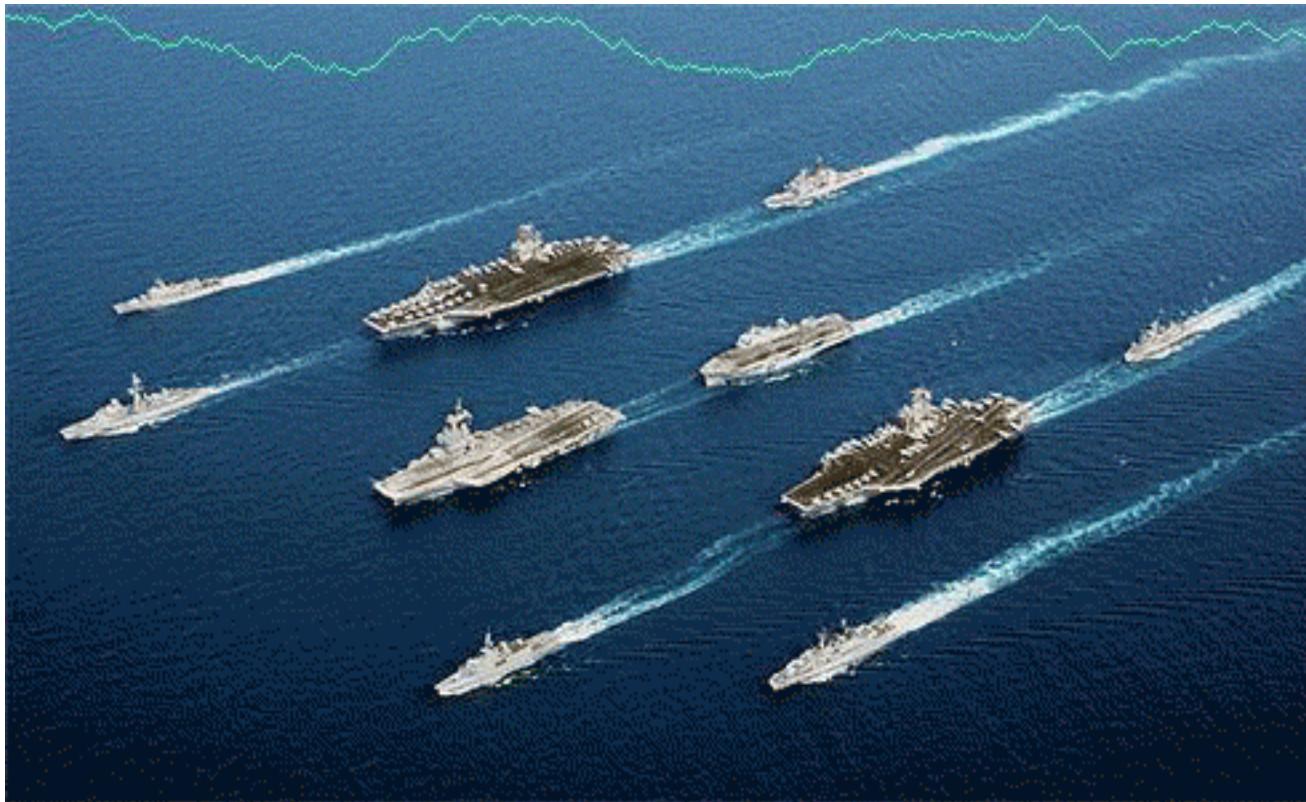


*Scaling*

# Example seam carving



# Another example



# Content Enhancement



How would you use seam carving to do this?

# Seam Carving in the Gradient Domain



# Questions?

- Q: Will the result be the same if the image is flipped upside down?
- A: Yes (up to numerical stability)
  
- Q: Can we improve the running time?
- A: Yes. by factor (account for locality of operations)

# A Local Operator



# Questions?

- Q: Will the result be the same if the image is flipped upside down?
- A: Yes (up to numerical stability)
- Q: Can we improve the running time?
- A: By factor (local operations)
- Q: What happens to the overall energy in the image during seam carving?

# Preserved Energy



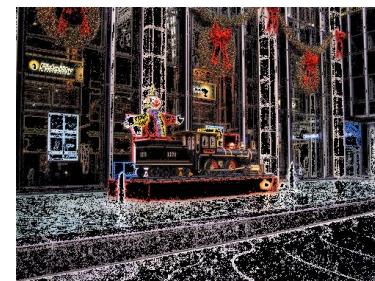
*Energy*



10%



30%



40%



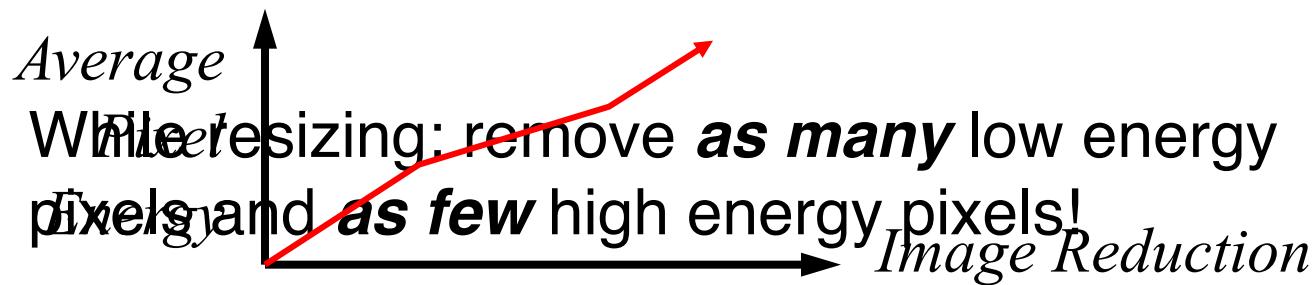
75%

While resizing: remove ***as many*** low energy pixels and ***as few*** high energy pixels!

# Preserved Energy

If we measure the average energy of pixels in the image after applying a resizing operator...

...the average should increase!



# Preserved Energy



Average  
Pixel  
Energy

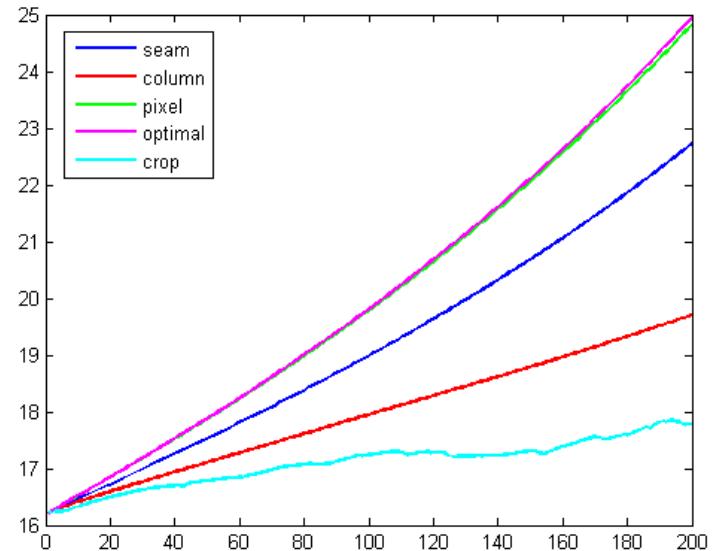
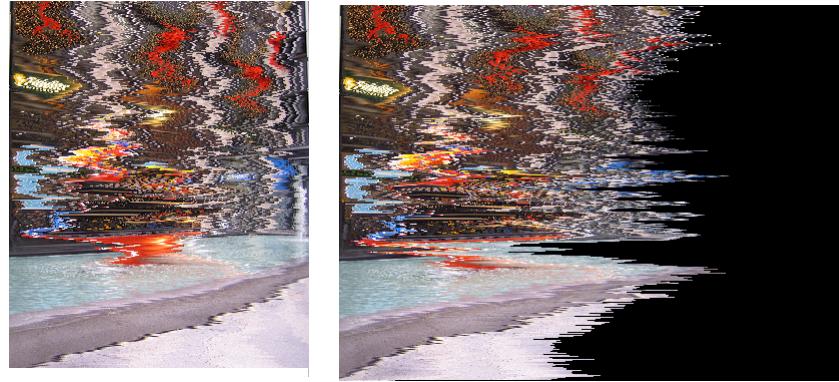


Image Reduction →



**crop**

**column**

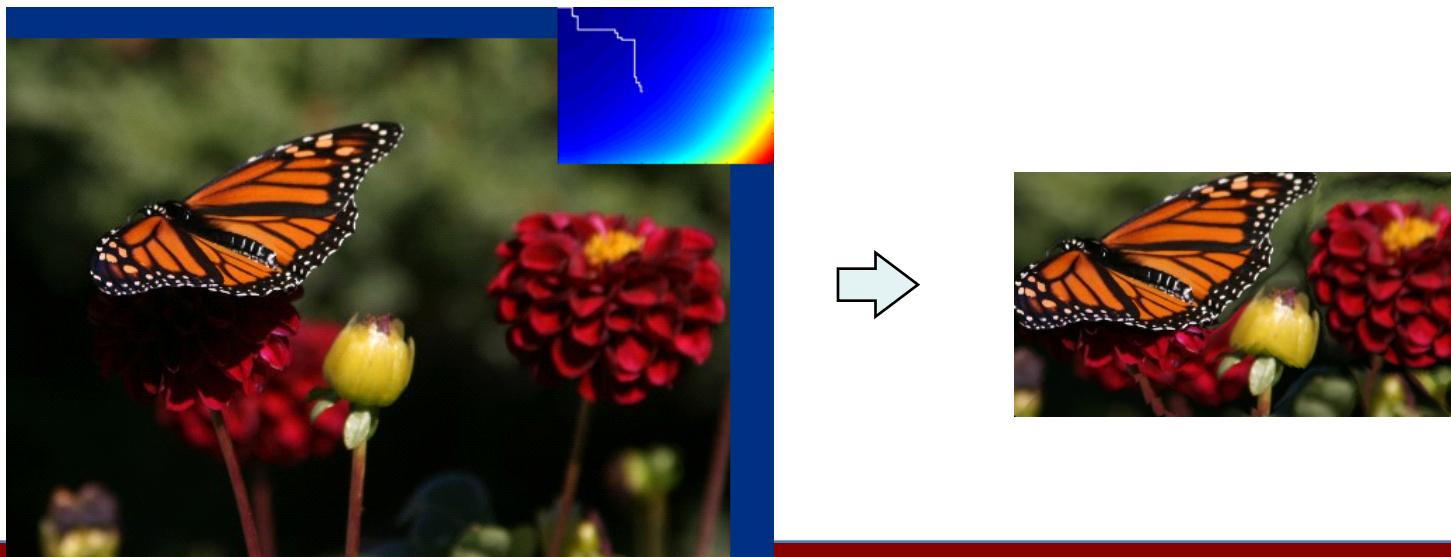
**seam**

**pixel**

**optimal**

# Both Dimensions?

- $m \times n \rightarrow m' \times n'$
- Remove horizontal seam first?
- Remove vertical seams first?
- Alternate between the two?
- The optimal order can be found! → Dynamic Prog (again)



# Retargeting in Both Dimensions

- The recursion relation:

$$T(r, c) = \min(T(r - 1, c) + E(s^x(I_{n-r-1 \times m-c})), T(r, c - 1) + E(s^y(I_{n-r \times m-c-1})))$$

$$\min_{s^x, s^y, \alpha} \sum_{i=1}^k E(\alpha_i s_i^x + (1 - \alpha_i) s_i^y)$$

# Retargeting in Both Dimensions

- The recursion relation:

$$T(r, c) = \min(T(r - 1, c) + E(s^x(I_{n-r-1 \times m-c})), T(r, c - 1) + E(s^y(I_{n-r \times m-c-1})))$$

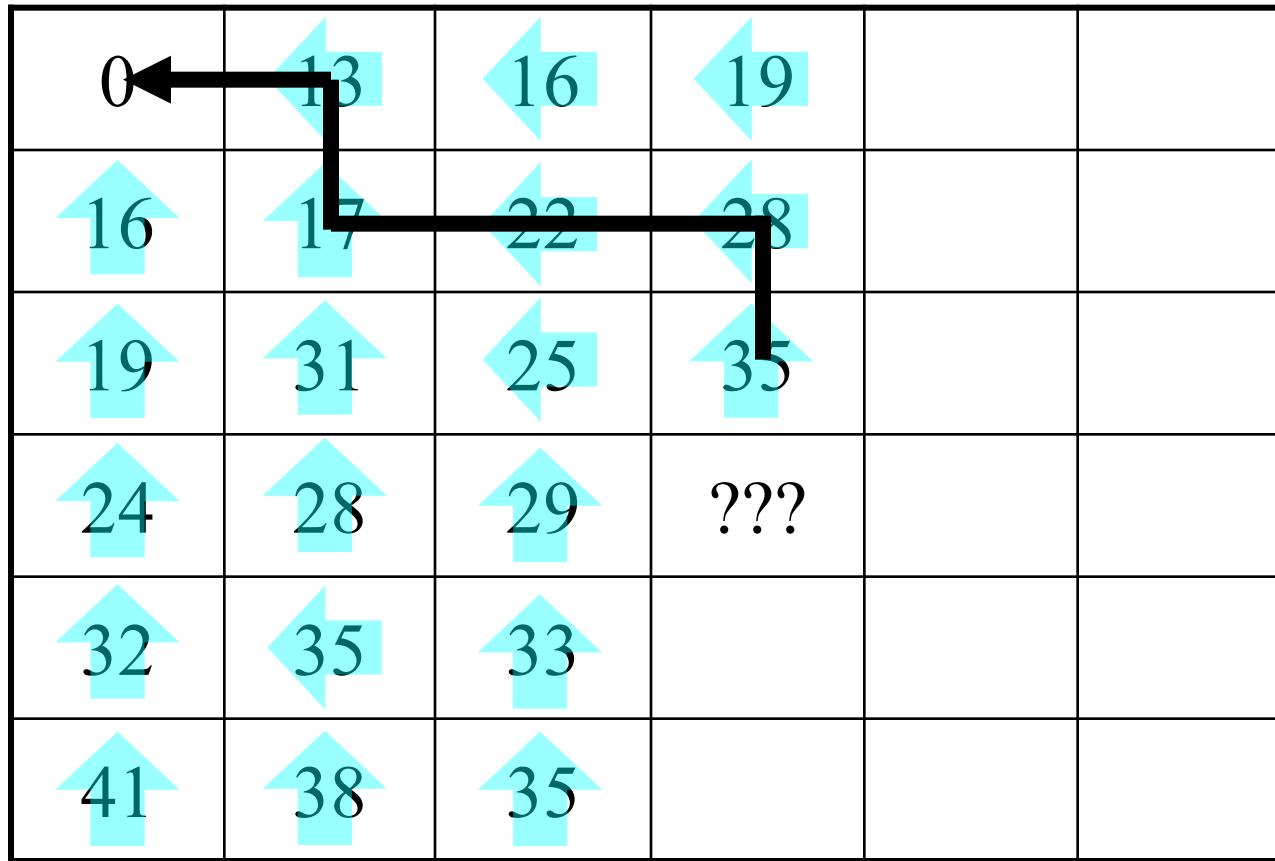
$$\min_{s^x, s^y, \alpha} \sum_{i=1}^k E(\alpha_i s_i^x + (1 - \alpha_i) s_i^y)$$



# Optimal Order Map

*Removal of vertical seams*

*Removal of horizontal seams*



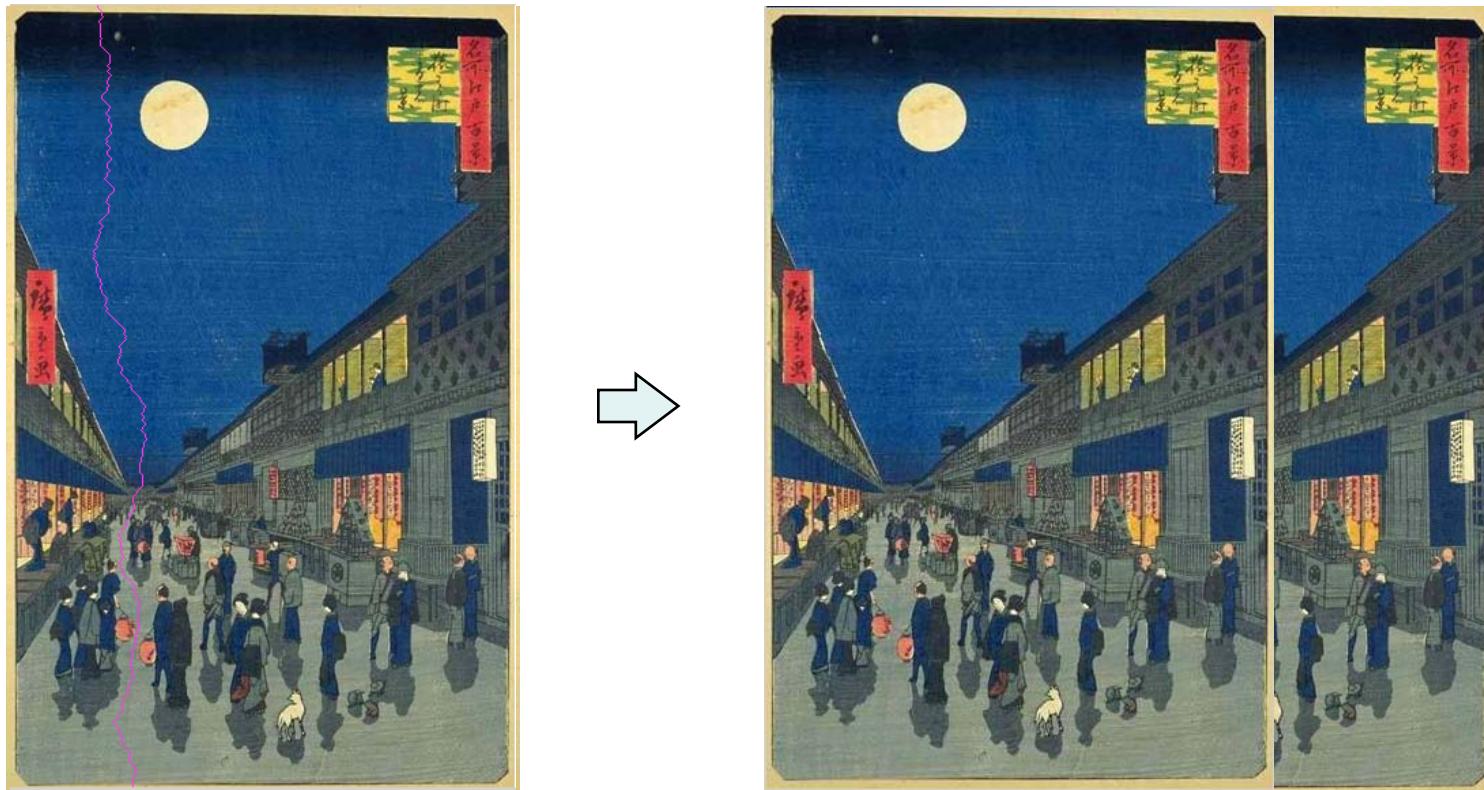
# Is it optimal...

- ... for removing ONE seam?
- ... for removing multiple seams?

# Is it optimal...

- ... for removing ONE seam?
- ... for removing multiple seams?
  - Consider HVV (how many possibly orderings?)
  - $\text{Cost}(V)$  on HV not necessarily equal  $\text{Cost}(V)$  on VH
  - But we keep track of only one:  $\min(HV, VH)\dots$

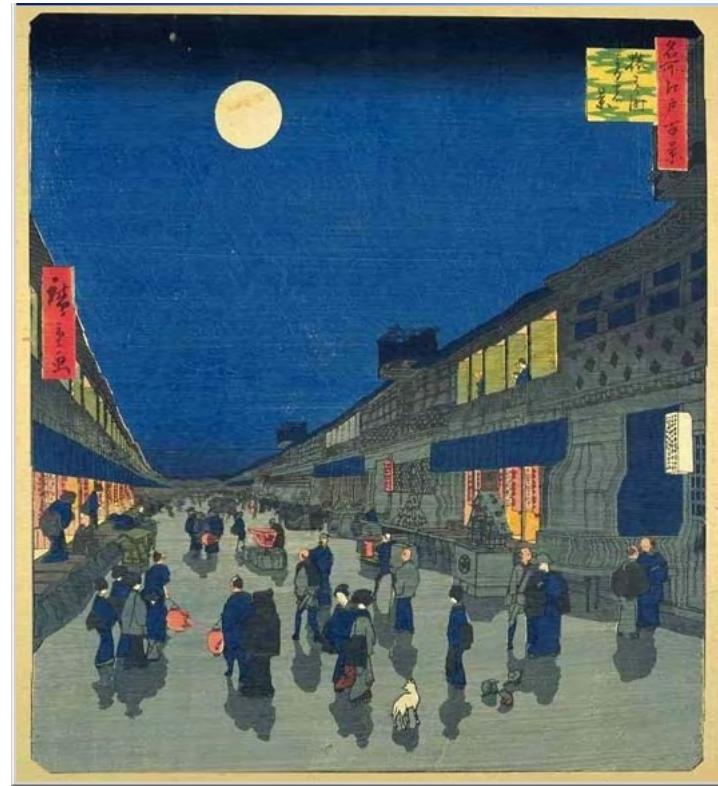
# Image Expansion (Synthesis)



# Image Expansion – take 2



Scaling



# Enlarged or Reduced?



# Combined Insert and Remove



*Insert & remove seams*



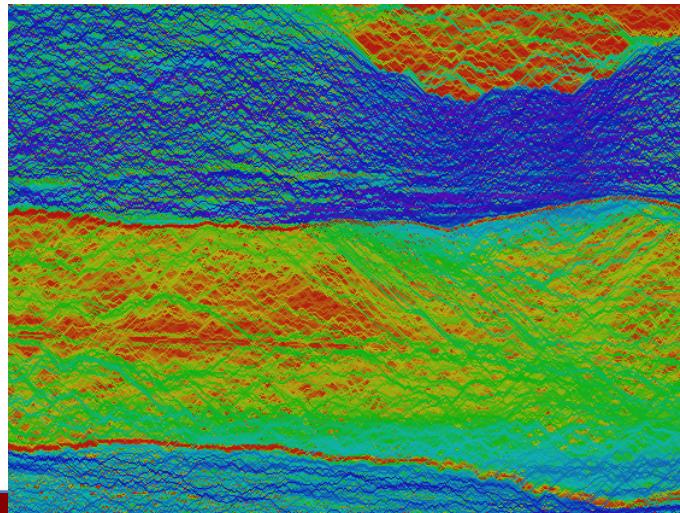
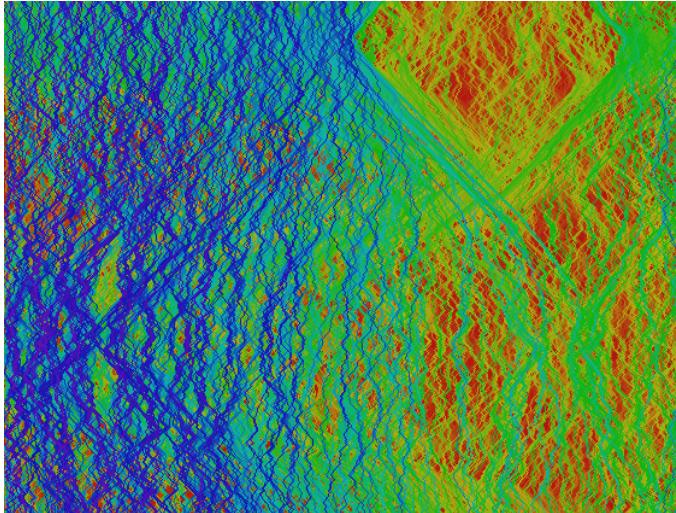
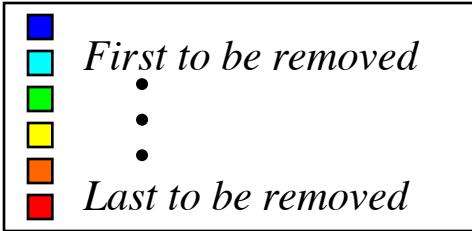
*Scaling*

# Questions?

# Multi-Size Images

- We can create a new *representation* of an image that will allow adapting it to different sizes!
  1. Precompute all seams once
  2. Realtime resizing / transmit with content

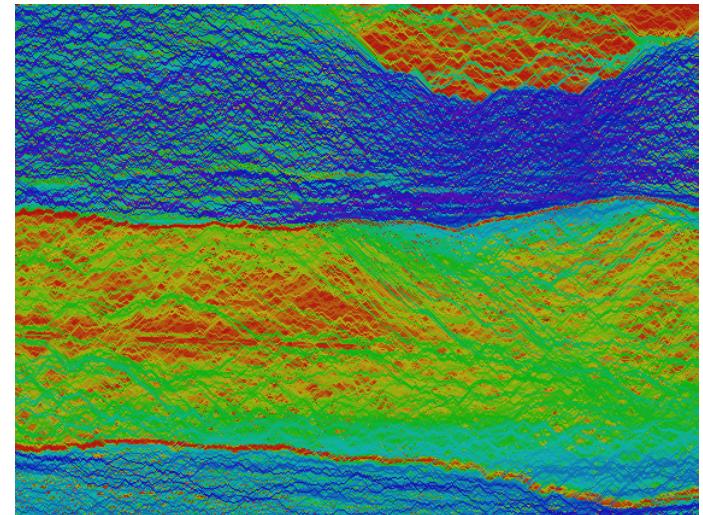
# Multi-Size Images



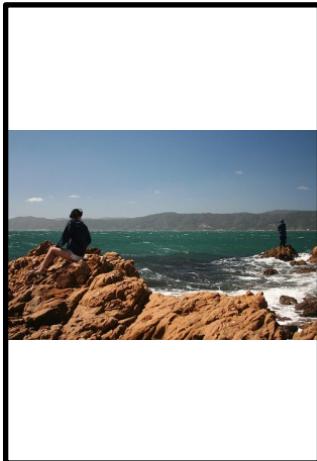
# Multi-Size Image Representation



+



# Multi-Size Image Representation



# 2D Multi-Size Representation?

- Alternatives
  - Use seams in one direction, row/column seams in the other direction
  - Compute seams in one direction and use them to constrain seams in the other direction
  - If you can do that you'll have a nice publication ☺

# Auxiliary Energy

- Recall our seam equation

$$\mathbf{M}(i, j) = E(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

# Object Removal



# Object Removal



Input

Retargeted

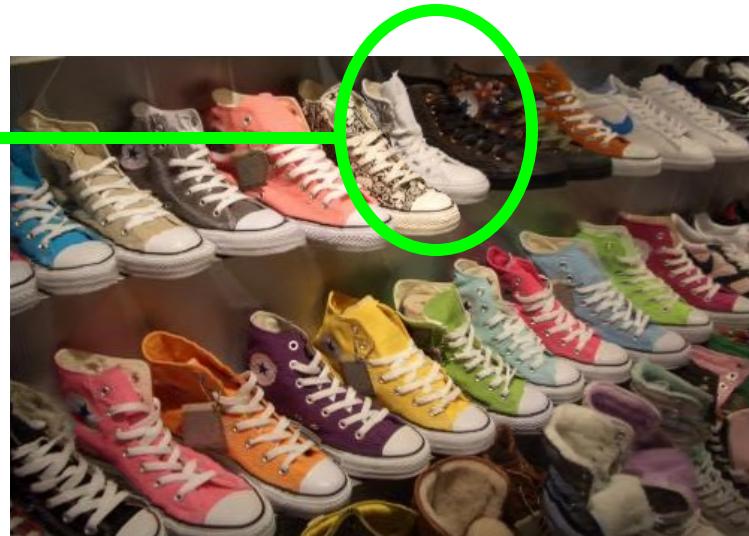
Pigeon Removed

Girl Removed

# Find the Missing Shoe!



# Solution



# Limitations

Content



Structure



# With face detector



# With User Constraints



# Preserved Energy - Revisited



Average  
Pixel  
Energy

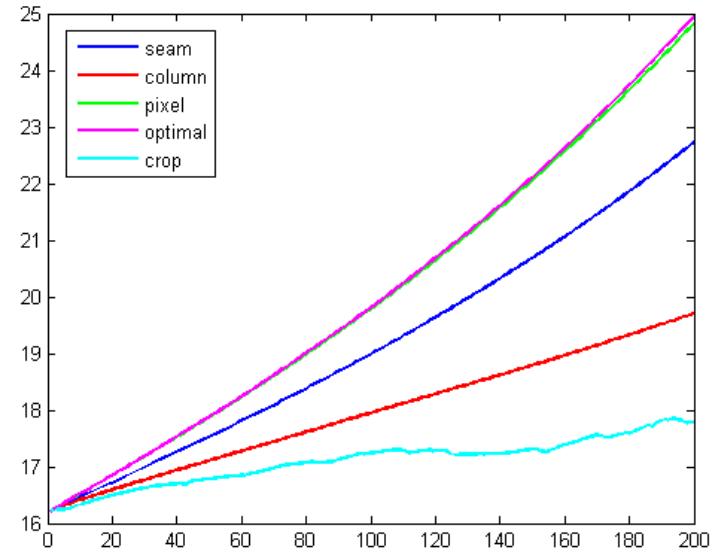
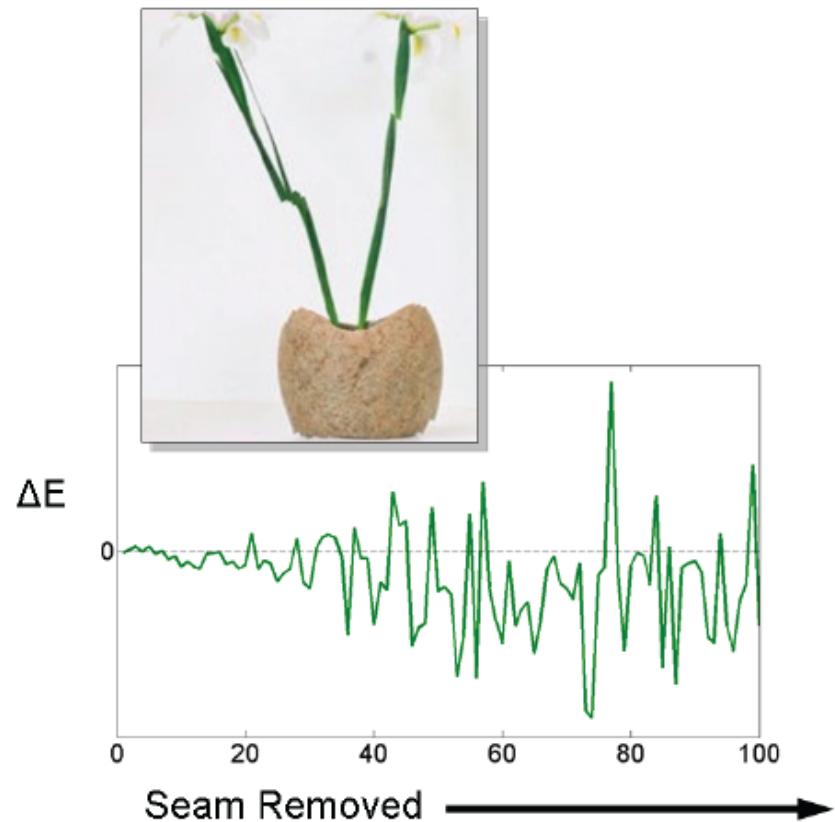
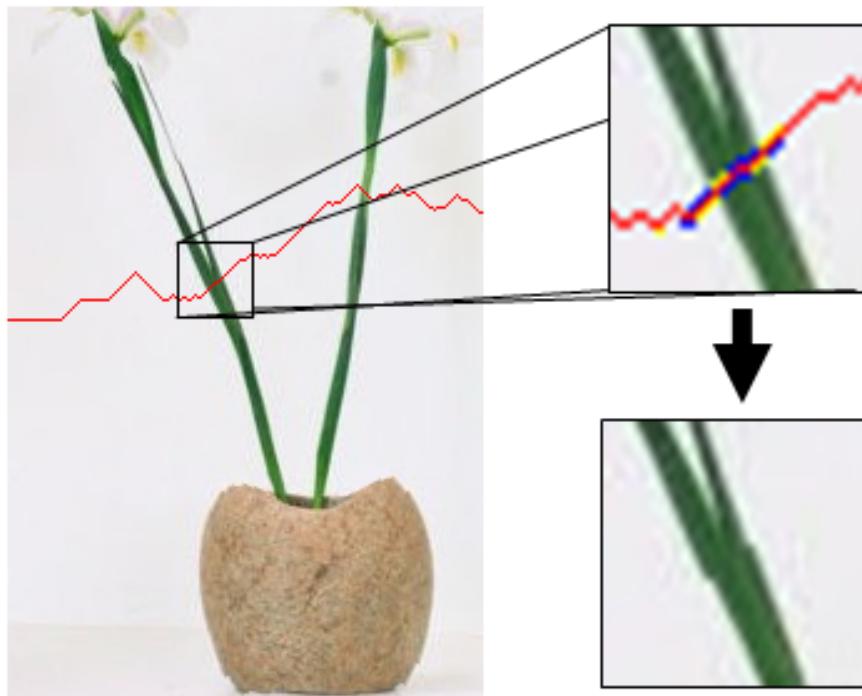


Image Reduction →



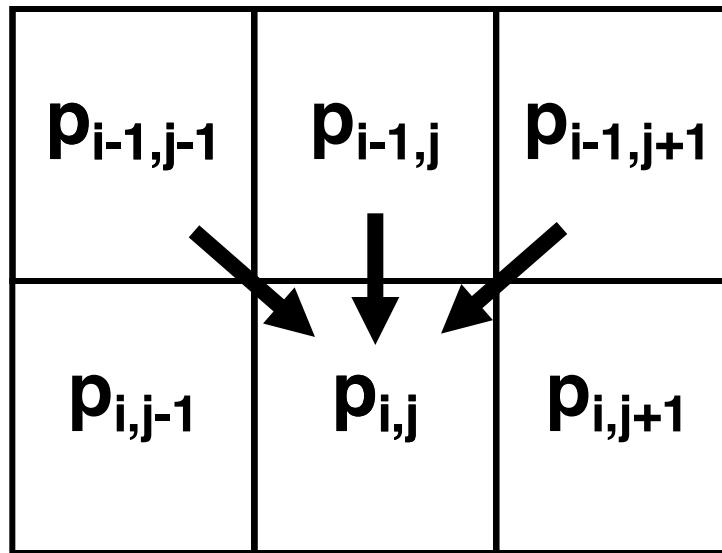
# Inserted Energy



# Minimize Inserted Energy

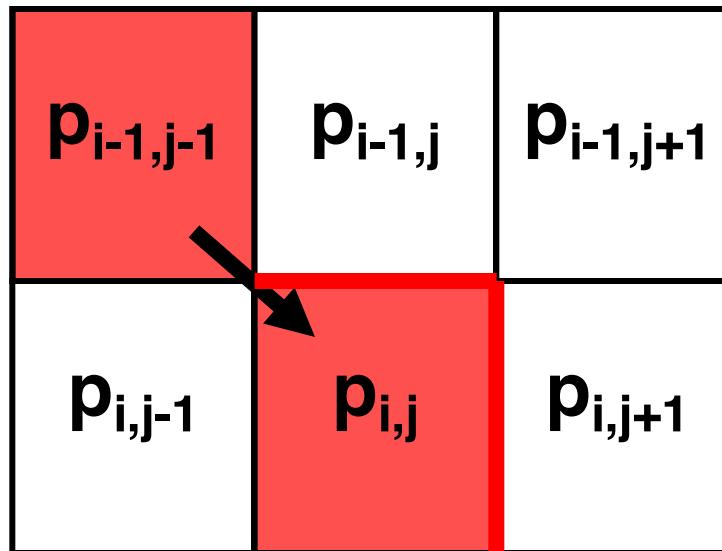
- Instead of removing the seam of least energy, remove the seam that *inserts the least energy* to the image !

# Tracking Inserted Energy



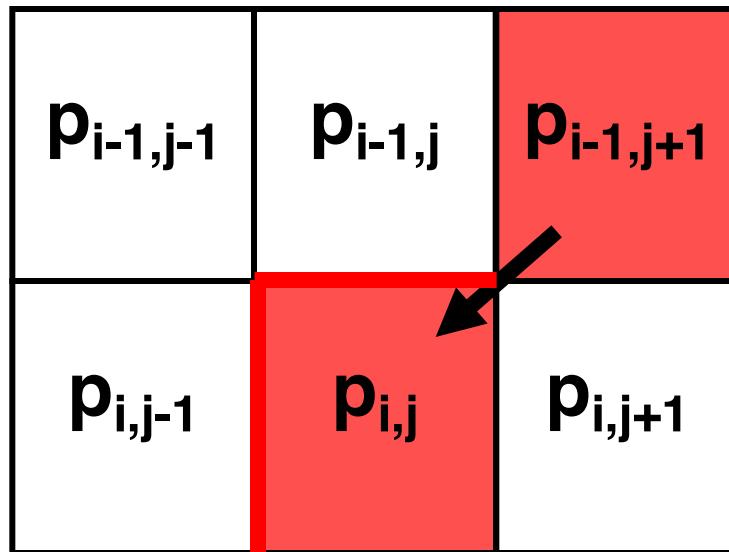
- Three possibilities when removing pixel  $P_{i,j}$

# Pixel $P_{i,j}$ : Left Seam



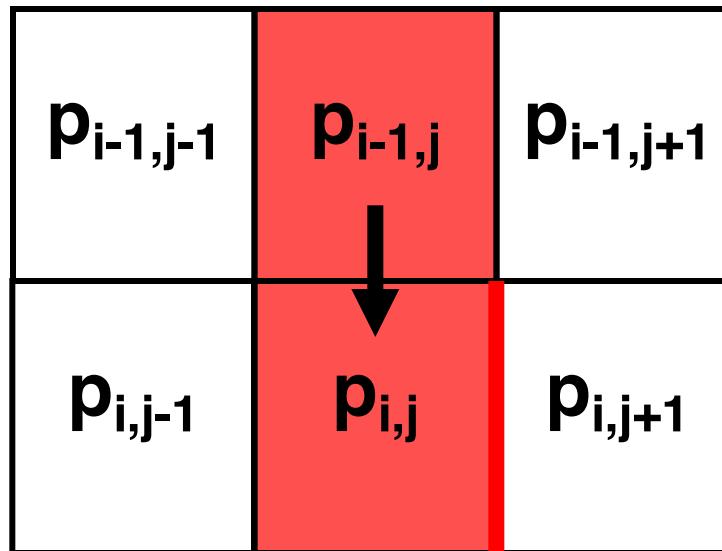
$$C_L(i, j) = |I(i, j + 1) - I(i, j - 1)| + |I(i - 1, j) - I(i, j - 1)|$$

# Pixel $P_{i,j}$ : Right Seam



$$C_R(i, j) = |I(i, j + 1) - I(i, j - 1)| + |I(i - 1, j) - I(i, j + 1)|$$

# Pixel $P_{i,j}$ : Vertical Seam



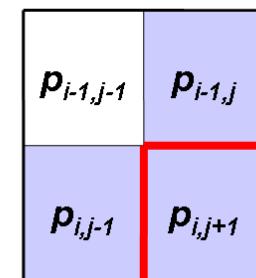
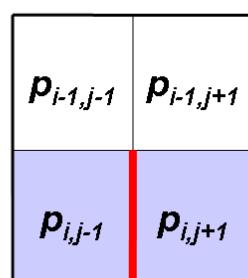
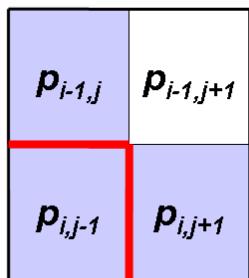
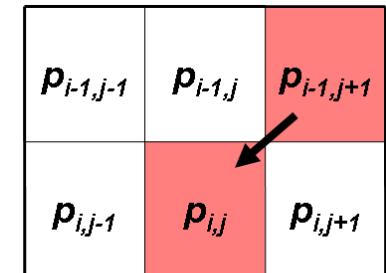
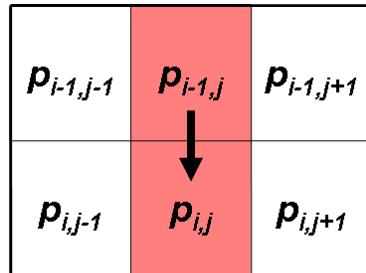
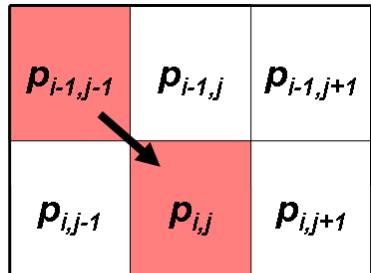
$$C_V(i, j) = |I(i, j + 1) - I(i, j - 1)|$$

# Old “Backward” Energy

$$M(i, j) = E(i, j) + \min \begin{cases} M(i - 1, j - 1) \\ M(i - 1, j) \\ M(i - 1, j + 1) \end{cases}$$

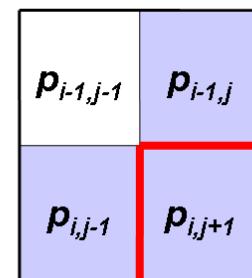
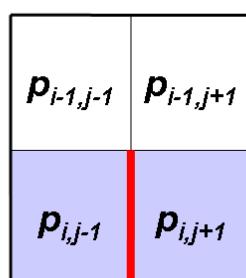
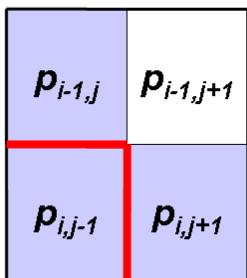
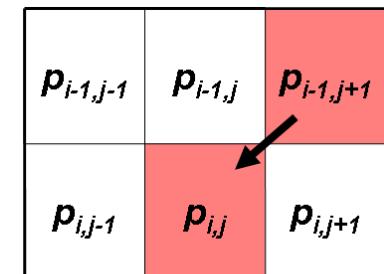
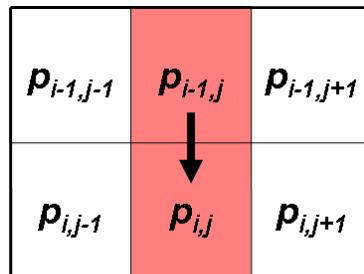
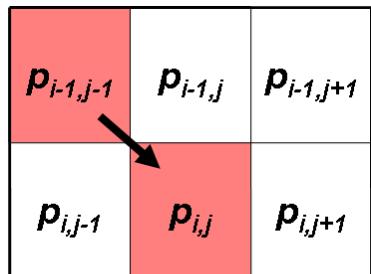
# New Forward Looking Energy

$$M(i, j) = \min \begin{cases} M(i - 1, j - 1) + C_L(i, j) \\ M(i - 1, j) + C_U(i, j), \\ M(i - 1, j + 1) + C_R(i, j) \end{cases}$$



# Adding “Pixel Energy”

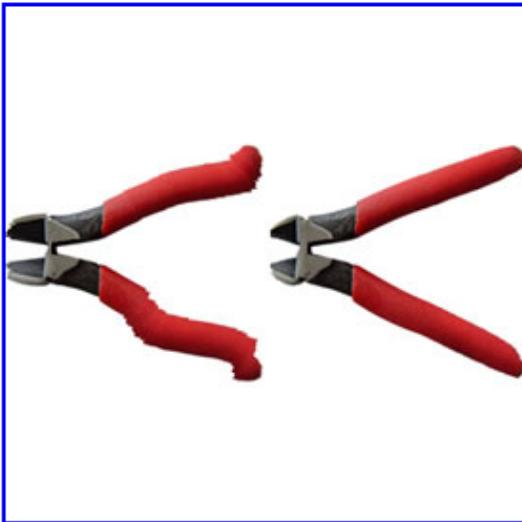
$$M(i, j) = P(i, j) + \min \begin{cases} M(i - 1, j - 1) + C_L(i, j) \\ M(i - 1, j) + C_U(i, j), \\ M(i - 1, j + 1) + C_R(i, j) \end{cases}$$



# Results



Input



Backward

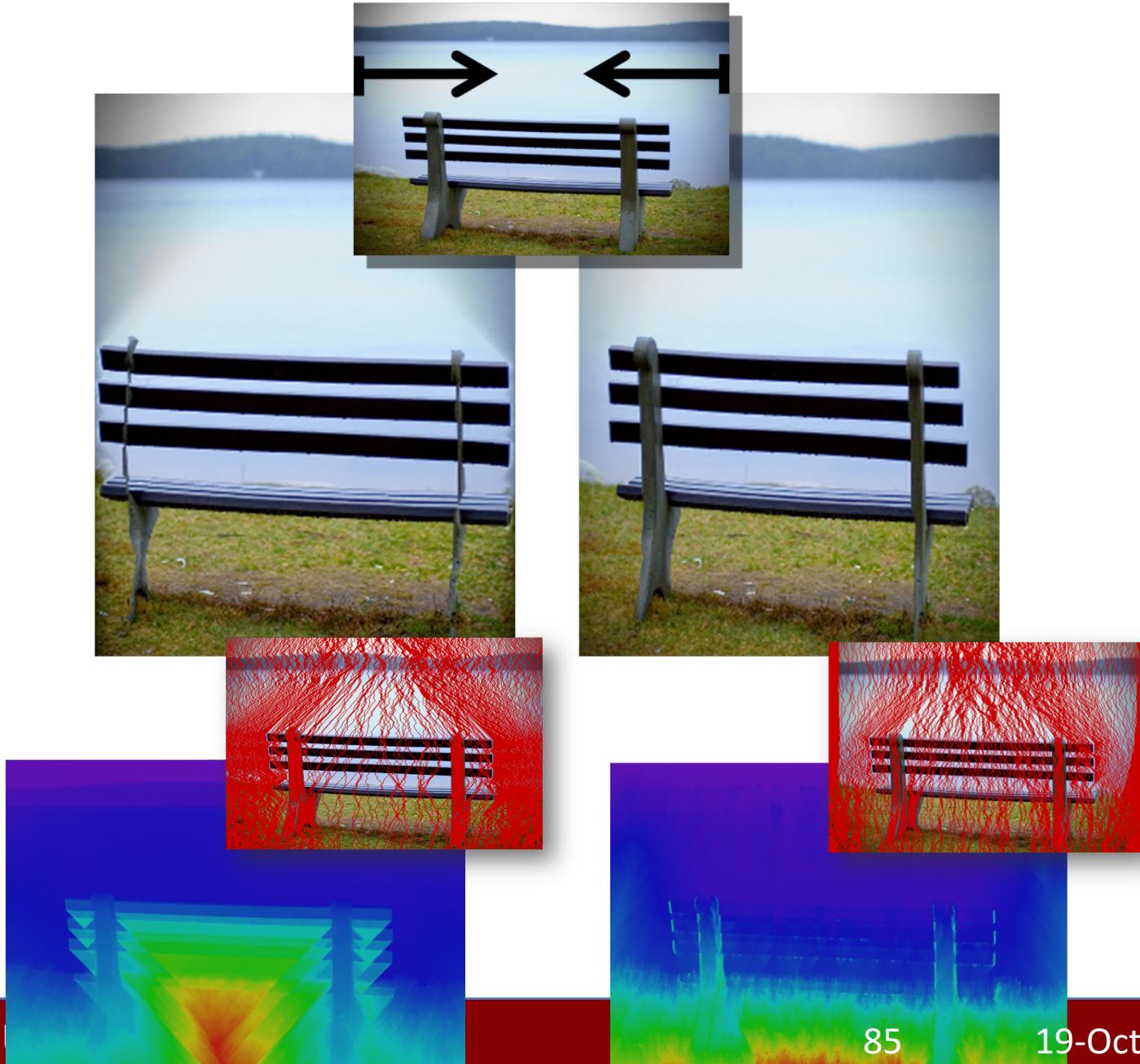
Forward



Input



Forward



# Backward vs. Forward

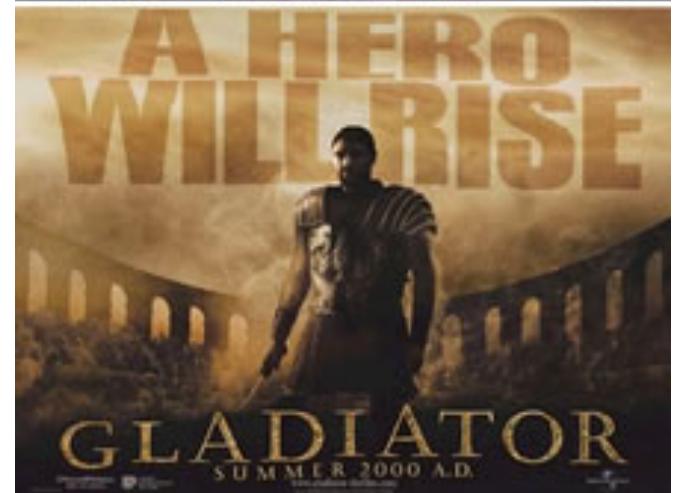
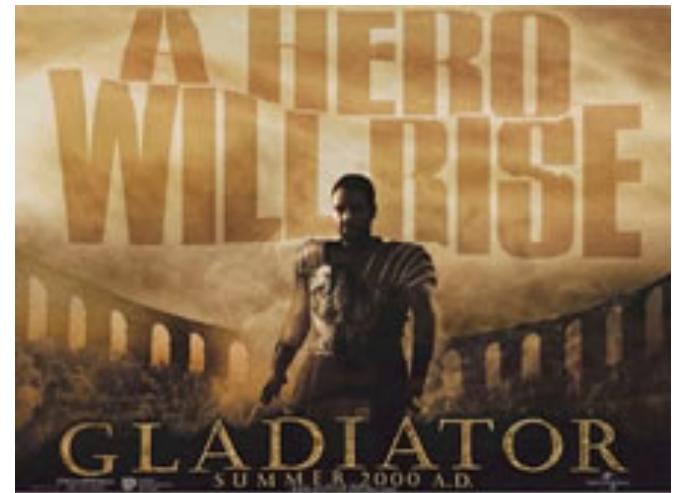


Backward



Forward

# Results

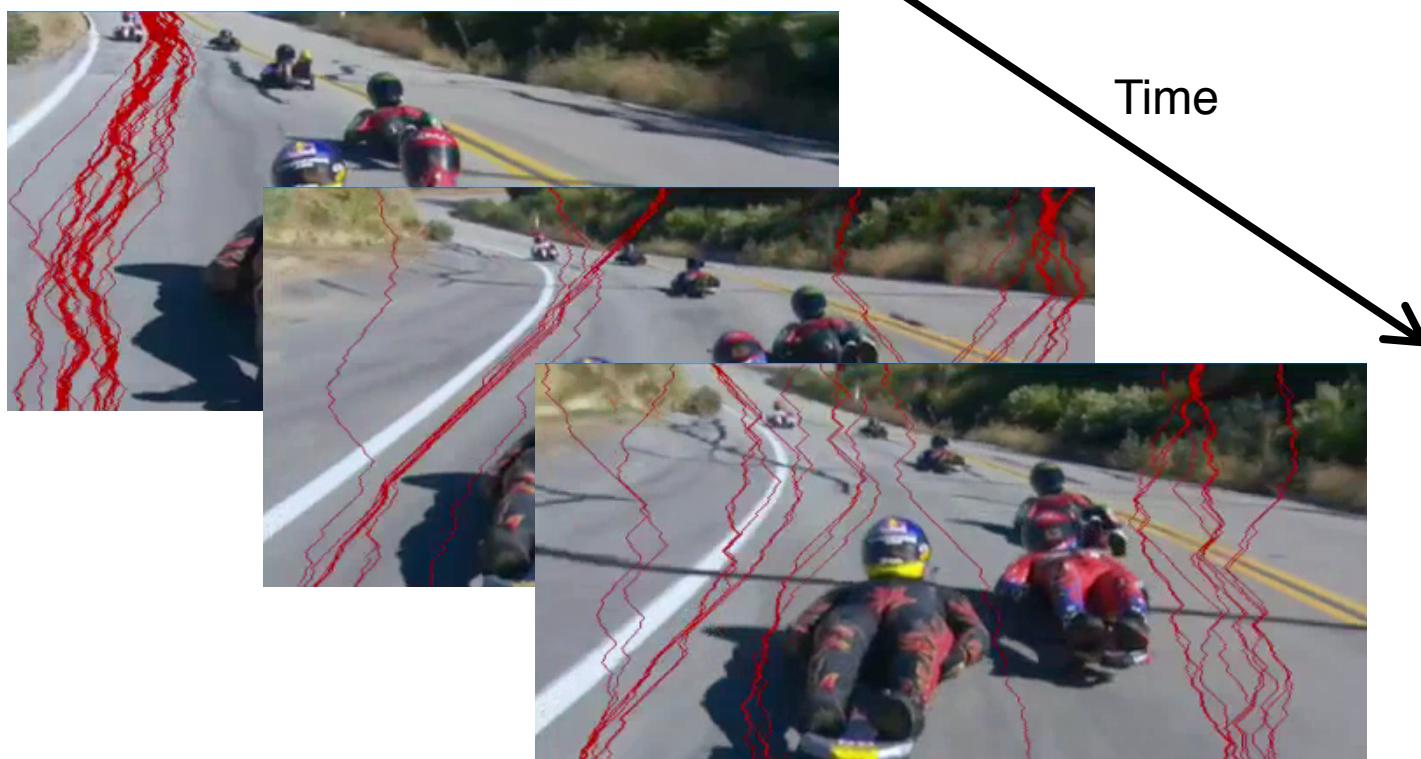


# From Images to Videos

- In general, video processing is a much (much!) harder problem
1. Cardinality
    - Suppose 1min of video x 30 fps = 1800 frames
    - Say your algorithm processes an image in 1 minute → **30 hours !!**
  2. Dimensionality/algorithmic
    - Temporal coherency: human visual system is highly sensitive to motion!

# Seam-Carving Video?

- Naive... frame by frame independently



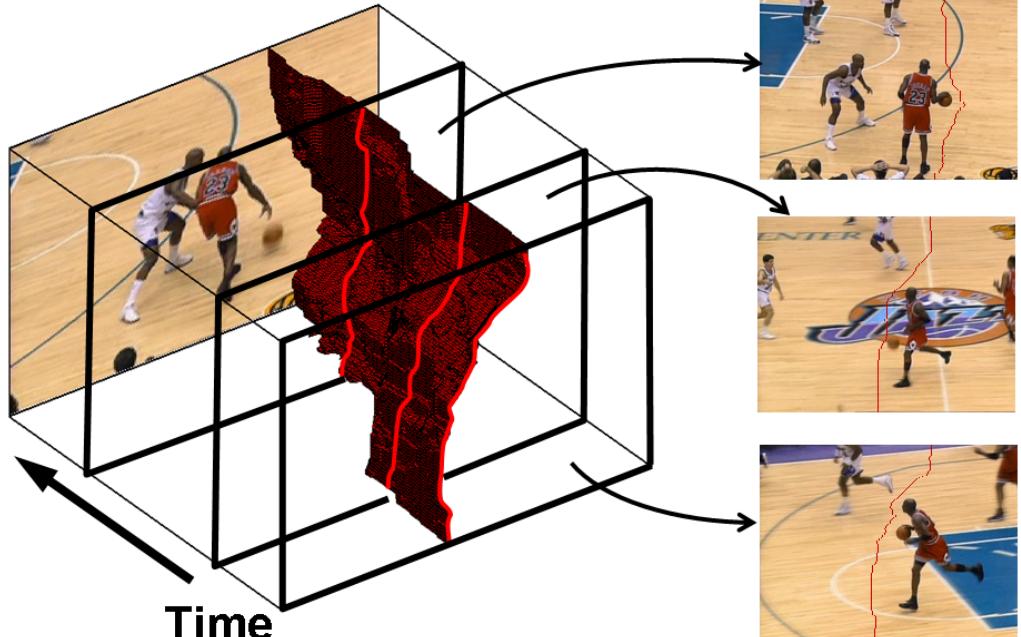
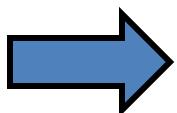
# Frame-by-frame Seam-Carving



# From 2D to 3D



1D paths in images



2D manifolds in video cubes

# Example video retargeting



# Object Removal...



# Object detection + seam carving



# References

- Seam Carving for Content-Aware Image Resizing – Avidan and Shamir 2007
- Content-driven Video Retargeting – Wolf et al. 2007
- Improved Seam Carving for Video Retargeting – Rubinstein et al. 2008
- *Optimized Scale-and-Stretch* for Image Resizing – Wang et al. 2008
- Summarizing Visual Data Using Bidirectional Similarity – Simakov et al. 2008
- Multi-operator Media Retargeting – Rubinstein et al. 2009
- Shift-Map Image Editing – Pritch et al. 2009
- Energy-Based Image Deformation – Karni et al. 2009
- Seam carving in Photoshop CS4: [http://help.adobe.com/en\\_US/Photoshop/11.0/WS6F81C45F-2AC0-4685-8FFD-DBA374BF21CD.html](http://help.adobe.com/en_US/Photoshop/11.0/WS6F81C45F-2AC0-4685-8FFD-DBA374BF21CD.html)